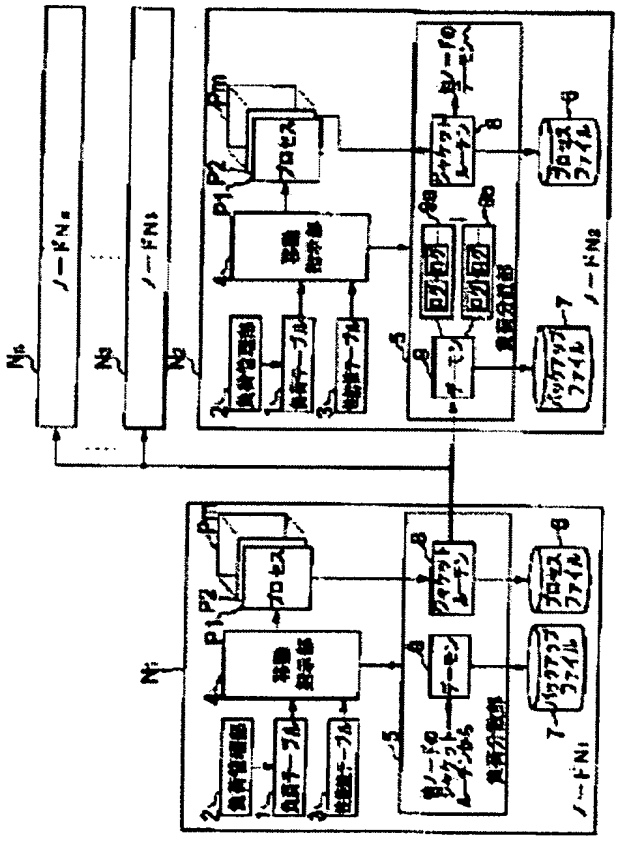
INTER-DISTRIBUTED NODE LOAD DISTRIBUTION SYSTEM

Publication number: JP2000137692
Publication date: 2000-05-16
Inventor: MURATA AKIFUMI
Applicant: TOKYO SHIBAURA ELECTRIC CO
Classification:
- International: ***G06F15/177; G06F9/50; G06F9/54;*** **G06F15/16; G06F9/46;** (IPC1-7): G06F15/177
- European:
Application number: JP19980311316 19981030
Priority number(s): JP19980311316 19981030

Report a data error here

Abstract of **JP2000137692**

PROBLEM TO BE SOLVED: To execute the distribution of loads when loads are centralized or distribution is requested without generating any waste in a memory resource. SOLUTION: This inter-node load distribution system is provided with a load table 1 for storing the present load value of each node N1-Nn and a performance value table 3 for storing the processing performance value of each node N1-Nn. At the time of detecting the excess load of its own node by referring to the load table 1, a movement instructing part 4 selects a node being the destination of movement which is capable of increasing loads by referring to the performance value table 3, and generates a load movement instruction. A load distributing part 5 moves the load of its own node to the node being the destination of movement for each prescribed unit based on the load movement instruction generated by the movement instructing part 4.



Data supplied from the ***esp@cenet*** database - Worldwide

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2000－137692
（P2000－137692A）
(43)公開日　平成12年５月16日(2000.5.16)

(51)Int.Cl.'	識別記号	ＦＩ		テーマコード'（参考）
Ｇ０６Ｆ　15/177	６７４	Ｇ０６Ｆ　15/177	６７４Ｂ	５Ｂ０４５
			６７４Ｃ	

審査請求　未請求　請求項の数８　ＯＬ　（全 10 頁）

(21)出願番号　特願平10－311316

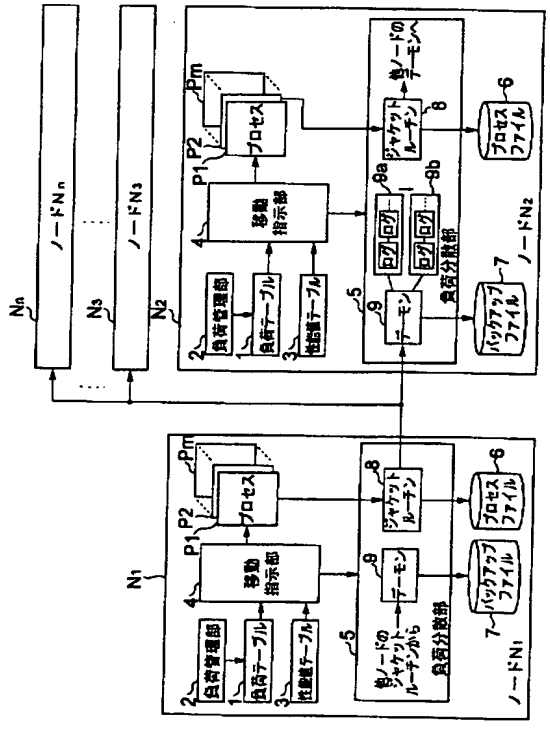
(22)出願日　平成10年10月30日(1998.10.30)

(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地
(72)発明者　村田　明文
東京都府中市東芝町１番地　株式会社東芝府中工場内
(74)代理人　100058479
弁理士　鈴江　武彦　（外６名）

Ｆターム(参考)　5B045 GG04

(54)【発明の名称】　分散ノード間負荷分散方式

(57)【要約】
【課題】　本発明は、メモリリソースに無駄を生じさせず、負荷集中時や分散要求時に負荷分散の実行を図る。
【解決手段】　各ノードＮ１～Ｎｎの現在の負荷値が記憶される負荷テーブル１と、各ノードＮ１～Ｎｎの処理性能値が記憶される性能値テーブル３とを有し、移動指示部４が、負荷テーブル１を参照して自ノードの過負荷を検出したとき、性能値テーブル３を参照して負荷を増加可能な移動先ノードを選択し、負荷移動指示を生成し、負荷分散部５が、移動指示部４により生成された負荷移動指示に基づいて、自ノードの負荷を所定単位毎に移動先ノードに移動させる分散ノード間負荷分散方式。



【特許請求の範囲】
【請求項１】　分散配置された複数のノードが互いに接続され、前記各ノードの有する負荷を各ノード間で分散させるための分散ノード間負荷分散方式であって、
前記各ノードの現在の負荷値が記憶される負荷記憶手段と、
前記各ノードの処理性能値が記憶される性能値記憶手段と、
前記負荷記憶手段を参照して自ノードの過負荷を検出したとき、前記性能値記憶手段を参照して負荷を増加可能な移動先ノードを選択し、負荷移動指示を生成する移動指示手段と、
前記移動指示手段により生成された負荷移動指示に基づいて、自ノードの負荷を所定単位毎に前記移動先ノードに移動させる負荷分散手段と
を備えたことを特徴とする分散ノード間負荷分散方式。
【請求項２】　請求項１に記載の分散ノード間負荷分散方式において、
前記移動指示手段は、予め複数の度合のいずれかに負荷が分類され、前記各度合毎に、移動対象の負荷が先頭にあり、移動された負荷が末尾に接続されるキューを備えたことを特徴とする分散ノード間負荷分散方式。
【請求項３】　請求項１に記載の分散ノード間負荷分散方式において、
前記負荷テーブルは、前記各負荷値と、前記各負荷値における時系列的な平均値と、前記各負荷値の平均２乗誤差とが記憶されており、
前記移動指示手段は、前記自ノードの過負荷を検出したとき、前記負荷テーブル内の平均値及び平均２乗誤差に基づいて、現在の負荷値と前記平均値とが所定値以上離れており、且つ前記平均２乗誤差の小さい負荷を移動対象に選択しないことを特徴とする分散ノード間負荷分散方式。
【請求項４】　請求項１に記載の分散ノード間負荷分散方式において、
前記移動指示手段は、移動先対象のノードが先頭にあり、移動先にされたノードが末尾に接続されるキューを備えたことを特徴とする分散ノード間負荷分散方式。
【請求項５】　請求項１に記載の分散ノード間負荷分散方式において、
前記負荷テーブルは、前記負荷値がプロセス毎に記憶され、且つ１個以上のプロセスからなる集合が前記所定単位として登録されていることを特徴とする分散ノード間負荷分散方式。
【請求項６】　請求項１に記載の分散ノード間負荷分散方式において、
前記性能値テーブルは、ＭＩＰＳ（１００万命令／秒）に基づいた前記処理性能値が記憶されることを特徴とする分散ノード間負荷分散方式。
【請求項７】　請求項１に記載の分散ノード間負荷分散方式において、
前記移動指示手段は、自ノードにおける移動対象の負荷の負荷値、前記移動先ノードの処理性能値及び前記自ノードの処理性能値に基づいて、前記移動対象の負荷を移動した場合に前記移動先ノードで増加する負荷値を算出し、前記増加する負荷値が前記移動先ノードでのＣＰＵのアイドル量よりも小のとき、前記負荷移動指示を生成することを特徴とする分散ノード間負荷分散方式。
【請求項８】　請求項１に記載の分散ノード間負荷分散方式において、
前記移動指示手段は、予め高負荷、中負荷又は小負荷のいずれかの度合に負荷が分類され、前記各度合毎に、移動対象の負荷が先頭にあり、移動された負荷が末尾に接続されるキューを備えたことを特徴とする分散ノード間負荷分散方式。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、分散ノードコンピューティング環境における分散ノード間負荷分散方式に関する。
【０００２】
【従来の技術】一般に、複数のノードが分散配置された分散ノード間では、複数のプロセス（サービス）を実行しつつ各ノードの負荷分散を行なう際の分散ノード間負荷分散方式が知られている。
【０００３】この種の分散ノード間負荷分散方式としては、例えば、ＲＰＣ（リモートプロシージャコール）によるＲＰＣサーバの分散方式がある。このＲＰＣサーバの分散方式は、予め負荷分散用に全サーバにＲＰＣサーバを立上げておき、全サーバに負荷を分散する方式である。また、分散ノード間負荷分散方式には、予めプロセス立上げ時に各サーバに負荷を分散する方式もある。
【０００４】
【発明が解決しようとする課題】しかしながら以上のような分散ノード間負荷分散方式では、例えばＲＰＣサーバの分散方式の場合、予め負荷分散用に全サーバにＲＰＣサーバを立上げる必要があるので、メモリリソースに無駄を生じさせる問題があり、また、負荷集中時又は分散要求時に負荷を分散し得ない問題がある。
【０００５】一方、プロセス立上げ時の分散方式の場合、負荷の集中したサーバに新たな負荷をかけないものの、この負荷の集中したサーバの負荷を分散し得ない問題がある。
【０００６】本発明は上記実情を考慮してなされたもので、メモリリソースに無駄を生じさせず、負荷集中時や分散要求時に負荷分散を実行し得る分散ノード間負荷分散方式を提供することを目的とする。
【０００７】
【課題を解決するための手段】請求項１に対応する発明は、分散配置された複数のノードが互いに接続され、前

記各ノードの有する負荷を各ノード間で分散させるための分散ノード間負荷分散方式であって、前記各ノードの現在の負荷値が記憶される負荷記憶手段と、前記各ノードの処理性能値が記憶される性能値記憶手段と、前記負荷記憶手段を参照して自ノードの過負荷を検出したとき、前記性能値記憶手段を参照して負荷を増加可能な移動先ノードを選択し、負荷移動指示を生成する移動指示手段と、前記移動指示手段により生成された負荷移動指示に基づいて、自ノードの負荷を所定単位毎に前記移動先ノードに移動させる負荷分散手段とを備えた分散ノード間負荷分散方式である。

【０００８】また、請求項２に対応する発明は、請求項１に対応する分散ノード間負荷分散方式において、前記移動指示手段としては、予め複数の度合のいずれかに負荷が分類され、前記各度合毎に、移動対象の負荷が先頭にあり、移動された負荷が末尾に接続されるキューを備えた分散ノード間負荷分散方式である。

【０００９】さらに、請求項３に対応する発明は、請求項１に対応する分散ノード間負荷分散方式において、前記負荷テーブルとしては、前記各負荷値と、前記各負荷値における時系列的な平均値と、前記各負荷値の平均２乗誤差とが記憶されており、前記移動指示手段としては、前記自ノードの過負荷を検出したとき、前記負荷テーブル内の平均値及び平均２乗誤差に基づいて、現在の負荷値と前記平均値とが所定値以上離れており、且つ前記平均２乗誤差の小さい負荷を移動対象に選択しない分散ノード間負荷分散方式である。

【００１０】また、請求項４に対応する発明は、請求項１に対応する分散ノード間負荷分散方式において、前記移動指示手段としては、移動先対象のノードが先頭にあり、移動先にされたノードが末尾に接続されるキューを備えた分散ノード間負荷分散方式である。

【００１１】さらに、請求項５に対応する発明は、請求項１に対応する分散ノード間負荷分散方式において、前記負荷テーブルとしては、前記負荷値がプロセス毎に記憶され、且つ１個以上のプロセスからなる集合が前記所定単位として登録されている分散ノード間負荷分散方式である。

【００１２】また、請求項６に対応する発明は、請求項１に対応する分散ノード間負荷分散方式において、前記性能値テーブルとしては、ＭＩＰＳ（１００万命令／秒）に基づいた前記処理性能値が記憶される分散ノード間負荷分散方式である。

【００１３】さらに、請求項７に対応する発明は、請求項１に対応する分散ノード間負荷分散方式において、前記移動指示手段としては、自ノードにおける移動対象の負荷の負荷値、前記移動先ノードの処理性能値及び前記自ノードの処理性能値に基づいて、前記移動対象の負荷を移動した場合に前記移動先ノードで増加する負荷値を算出し、前記増加する負荷値が前記移動先ノードでのＣＰＵのアイドル量よりも小のとき、前記負荷移動指示を生成する分散ノード間負荷分散方式である。

【００１４】また、請求項８に対応する発明は、請求項１に対応する分散ノード間負荷分散方式において、前記移動指示手段としては、予め高負荷、中負荷又は小負荷のいずれかの度合に負荷が分類され、前記各度合毎に、移動対象の負荷が先頭にあり、移動された負荷が末尾に接続されるキューを備えた分散ノード間負荷分散方式である。

（作用）従って、請求項１に対応する発明は以上のような手段を講じたことにより、各ノードの現在の負荷値が記憶される負荷記憶手段と、各ノードの処理性能値が記憶される性能値記憶手段とを有し、移動指示手段が、負荷記憶手段を参照して自ノードの過負荷を検出したとき、性能値記憶手段を参照して負荷を増加可能な移動先ノードを選択し、負荷移動指示を生成し、負荷分散手段が、移動指示手段により生成された負荷移動指示に基づいて、自ノードの負荷を所定単位毎に移動先ノードに移動させるので、従来とは異なり、メモリリソースに無駄を生じさせず、負荷集中時や分散要求時に負荷分散を実行させることができる。

【００１５】また、請求項２に対応する発明は、移動指示手段としては、予め複数の度合のいずれかに負荷が分類され、各度合毎に、移動対象の負荷が先頭にあり、移動された負荷が末尾に接続されるキューを備えたので、請求項１に対応する作用に加え、特定の負荷のみが順番に移動するたらい回し動作を阻止することができる。

【００１６】さらに、請求項３に対応する発明は、負荷テーブルとしては、各負荷値と、各負荷値における時系列的な平均値と、各負荷値の平均２乗誤差とが記憶されており、移動指示手段としては、自ノードの過負荷を検出したとき、負荷テーブル内の平均値及び平均２乗誤差に基づいて、現在の負荷値と平均値とが所定値以上離れており、且つ平均２乗誤差の小さい負荷を移動対象に選択しない。

【００１７】これにより、請求項１に対応する作用に加え、通常は低負荷値で一時的に高負荷値となる負荷の移動を阻止できるので、移動によるオーバヘッドを抑制することができる。

【００１８】また、請求項４に対応する発明は、移動指示手段としては、移動先対象のノードが先頭にあり、移動先にされたノードが末尾に接続されるキューを備えたので、請求項１に対応する作用に加え、負荷の移動先を選択する際に、他の全ノードのうち、一部のノードを検索すればよいので、サービス移動のオーバヘッドを抑制することができる。

【００１９】さらに、請求項５に対応する発明は、負荷テーブルとしては、負荷値がプロセス毎に記憶され、且つ１個以上のプロセスからなる集合が所定単位として登録されているので、請求項１に対応する作用を容易且つ

確実に奏することができる。

【００２０】また、請求項６に対応する発明は、性能値テーブルとしては、ＭＩＰＳ（１００万命令／秒）に基づいた処理性能値が記憶されるので、請求項１に対応する作用を容易且つ確実に奏することができる。

【００２１】さらに、請求項７に対応する発明は、移動指示手段としては、自ノードにおける移動対象の負荷の負荷値、移動先ノードの処理性能値及び自ノードの処理性能値に基づいて、移動対象の負荷を移動した場合に移動先ノードで増加する負荷値を算出し、増加する負荷値が移動先ノードでのＣＰＵのアイドル量よりも小のとき、負荷移動指示を生成するので、請求項１に対応する作用を容易且つ確実に奏することができる。

【００２２】また、請求項８に対応する発明は、移動指示手段としては、予め高負荷、中負荷又は小負荷のいずれかの度合に負荷が分類され、各度合毎に、移動対象の負荷が先頭にあり、移動された負荷が末尾に接続されるキューを備えたので、請求項１に対応する作用に加え、特定の負荷のみが順番に移動するたらい回し動作を阻止することができる。

【００２３】

【発明の実施の形態】以下、本発明の各実施形態について図面を参照しながら説明する。

（第１の実施形態）図１は本発明の第１の実施形態に係る分散ノード間負荷分散方式の適用された計算機システムの構成を示す模式図である。この計算機システムは、複数のノード（計算機本体）Ｎ１～Ｎｎが互いに接続されている。ここで、各ノードＮ１～Ｎｎは、実行中のプロセス（プログラム）が異なるものの、互いに同一構成のため、ノードＮ１を例に挙げて説明する。

【００２４】ノードＮ１は、実行の有無によらずに保持する複数のプロセスＰ１～Ｐｍの他、負荷テーブル１、負荷管理部２、性能値テーブル３、移動指示部４、負荷分散部５、プロセスファイル６及びバックアップファイル７を備えている。

【００２５】負荷テーブル１は、図２に示すように、負荷管理部２によって、各ノードＮ１～ＮｎにおけるプロセスＰ１～Ｐｍ毎の現在のＣＰＵ負荷値、メモリ負荷値及びディスク負荷値が読出／書込可能に記憶されるテーブルであり、１以上の任意のプロセスＰをまとめた集合が、１つのサービスＳ（負荷分散の単位）として取扱われる。

【００２６】なお、負荷テーブル１は、以上の値に加え、サービスＳ１～Ｓｊ毎の負荷値（各サービス内の各プロセス負荷の合計値）が記憶されてもよく、ノードＮ１～Ｎｎ毎の負荷値（ノード内の各サービス負荷の合計値）が記憶されてもよい。

【００２７】また、ノードＮ１全体のメモリ負荷値は、ノードＮ１のスワップ(swap)の使用量（スワップの残り容量あるいは一定時間のスワップin/out、ページin/outの量）で規定可能である。ノードＮ１全体のＣＰＵ負荷値は、ＣＰＵのアイドル量あるいはＣＰＵ割当て待ちプロセス数で規定可能である。

【００２８】負荷管理部２は、定期的に各ノードＮ１～ＮｎにおけるプロセスＰ１～Ｐｍ毎の現在のＣＰＵ負荷値、メモリ負荷値及びディスク負荷値を収集し、これらＣＰＵ負荷値、メモリ負荷値及びディスク負荷値を負荷テーブル１に書込む機能をもっている。

【００２９】性能値テーブル３は、ノードＮ１からＮｎ毎に予めＣＰＵ性能値及びメモリ性能値が読出可能に登録されたテーブルであり、ノードＮ１～Ｎｎ内の共有領域（共有メモリでもファイルでも可）に設けられている。ここで、ＣＰＵ性能値としては、１／（ＭＩＰＳ＊ＭＰＵ）が使用可能となっている。なお、ＭＩＰＳ（１００万命令／秒）は、１つのＣＰＵの性能値であり、ＭＰＵは、ＣＰＵの個数である。一方、メモリ性能値としては、ノード内の各メモリ容量の合計値が使用可能となっている。

【００３０】移動指示部４は、定期的に負荷テーブル１を参照して自ノードＮ１の負荷状況を調査し、負荷が所定値を越えた旨（過負荷）を検出すると、自ノードＮ１の各プロセスＰをサービスＳ単位で低負荷のノードＮｉ（ｉは１～ｎまでの任意の自然数（但し、自ノードの番号を除く））に移動させる旨の指示を負荷分散部５に与える機能をもっている。

【００３１】同様に、移動指示部４は、ノードＮ自体又はプロセスＰの障害発生時に、障害発生により実行不可能となったプロセスＰを含むサービスＳを低負荷のノードＮｉに移動させる旨の指示を負荷分散部５に与える機能をもっている。

【００３２】負荷分散部５は、移動指示部４から受けた負荷移動指示に基づいて、自ノードＮ１の移動対象の負荷（プロセス）をサービスＳ単位で他のノードＮｉに移動させる移動機能をもっている。

【００３３】ここで、移動機能は、自ノードＮ１に移動対象のプロセスＰｋ（ｋは１～ｍまでの任意の自然数）があるとき、自ノードＮ１における移動対象のプロセスＰｋを停止させ（障害により既に停止していれば不要）、移動対象のプロセスＰｋを立上げるための再開実行指示を移動先の他ノードＮｉに与え、他ノードＮｉにおける移動対象のプロセスＰｋを再開実行させることにより、結果としてプロセスＰｋを自ノードＮ１から他ノードＮｉに移動させるものである。

【００３４】この移動機能として、負荷分散部５は、移動指示部４から受ける指示により、実行途中のプロセスＰをサービスＳ単位で他ノードＮｉへ移送して低負荷の他ノードＮｉで継続実行する技術（特願平９－２３２９３０号）を用いている。

【００３５】係る技術を用いる負荷分散部５は、プロセス実行により更新されるプロセスファイル６の更新内容

の記録（以下、ログという）を採取して他の全ノードＮ２～Ｎｎに分散するためのジャケットルーチン８と、ジャケットルーチン８から受けたログを未確定キュー９ａとして保持すると共に、チェックポイント毎に未確定キュー９ａを確定キュー９ｂとして該確定キュー９ｂ内の各ログに基づいてプロセスのバックアップファイル７を更新可能なデーモン９とを備えている。

【００３６】ここで、ログは、プロセス状態を示すものであり、例えば、データ等のレジスタ情報及びファイルの更新等のシステムコール発行結果が使用可能である。チェックポイントとしては、一定時間が経過した時点、あるいはＯＳのシステムコール発行コールをフェッチした時点が使用可能である。

【００３７】次に、以上のように構成された計算機システムの動作を説明する。ノードＮ１における負荷分散部５のジャケットルーチン８は、図３に示すように、実行中のプロセスＰ１から各ログを採取し、これら各ログをプロセスファイルに更新記憶させると共に少なくともチェックポイントＣＰまでに他の各ノードＮ２～Ｎｎに送信する。各ノードＮ２～Ｎｎでは、デーモン９がこのログを受けて未確定キューとして保持し、チェックポイントＣＰ毎にログ内のシステムコール発行結果を反映させて処理を実行する。例えばシステムコール発行結果がバックアップファイル７の更新を示すとき、デーモン９によりバックアップファイル７を更新する。

【００３８】一方、ノードＮ１では、移動指示部４が、定期的に負荷テーブル１の情報を監視し、一定条件（自ノードＮ１の負荷が所定値を越えた時点）を満たすと、負荷分散を開始する。

【００３９】すなわち、移動指示部４は、図４に示すように、一定時間スリープ(sleep)し（ＳＴ１）、しかる後、負荷テーブル１を参照して自ノードＮ１の負荷状況を調査する（ＳＴ２）。

【００４０】この調査において、自ノードＮ１について過負荷か否かを判定し（ＳＴ３）、過負荷でないときにはステップＳＴ１へ戻る。

【００４１】なお、ステップＳＴ３の判定は、メモリ負荷の場合、前述したノードＮ１全体のメモリ負荷値が所定値を越えたときに過負荷とし、ＣＰＵ負荷の場合、前述したノードＮ１全体のＣＰＵ負荷値が所定値を越えたときに過負荷とする。

【００４２】自ノードＮ１を過負荷と判定したとき、自ノードＮ１から高負荷のサービスＳを他ノードＮｉへの転送対象として選択する（ＳＴ４）。

【００４３】続いて、最低の負荷の例えばノードＮ２を選択し（ＳＴ５）、負荷テーブル１及び性能値テーブル３を参照しつつ、サービスＳを移動可能か否かを判定する（ＳＴ６）。

【００４４】ステップＳＴ６の判定は、メモリ負荷と、ＣＰＵ負荷との２通りが実行される。メモリ負荷の判定は、移動先ノードＮ２のメモリ性能値をMtotal とし、移動先ノードＮ２で現在使用中のメモリ負荷値をMusing とし、移動するサービスＳのメモリ負荷値をMservとした場合、次の（１）式を満たすときに移動可能とされる。

Mtotal － Musing ＞ Mserv …（１）

なお、メモリ性能値はノードＮ２内の各メモリ容量の合計値である。

【００４５】次に、ＣＰＵ負荷の判定は、移動先ノードＮ２の現在のＣＰＵアイドル（遊休）量を先ＣＰＵidolとし、移動前のノードＮ１でのサービスＳのＣＰＵ負荷値を前ＣＰＵloadとし、移動先ノードＮ２のＣＰＵ性能値を先ＣＰＵperfとし、移動前のノードＮ１でのＣＰＵ性能値を前ＣＰＵperfとした場合、次の（２）式を満たすときに移動可能とされる。

【００４６】

【数１】

$$\text{先CPUidol} > \text{前CPUload} \times \text{先CPUperf} / \text{前CPUperf} \quad \cdots (2)$$

なお、ＣＰＵアイドル量は、新たに使用可能なＣＰＵ負荷値を意味している。すなわち、（２）式は、右辺の移動前のＣＰＵ負荷から換算される移動先のＣＰＵ負荷よりも、左辺の移動先のＣＰＵアイドル量が大である関係を意味している。

【００４７】また、サービス移動後の移動先ノードＮｉの評価において、ＣＰＵアイドル量を後ＣＰＵidolとし、メモリ負荷値を後Muse とした場合、各ノードＮ２～Ｎｎのうち、次の（３）式の値が最高のノードＮ２が、最低の負荷のノードＮ２として判定される。

【００４８】

【数２】

$$\text{後CPUidol} + 100 \times (\text{Mtotal} - \text{後Muse}) / \text{Mtotal} \quad \cdots (3)$$

すなわち、（３）式は、サービス移動後において、ＣＰＵ負荷の余裕分と、メモリ負荷の余裕分とを合計した値を示している。

【００４９】ステップＳＴ６においては、（１）式，（２）式を共に満たした場合、すなわち、メモリ負荷及びＣＰＵ負荷を共に移動可能と判定したときのみ、サービスＳを移動可能と判定し、負荷分散部５にサービス移動の指示を出し（ＳＴ７）、ステップＳＴ１へ戻る。

【００５０】なお、ステップ６において、高負荷のサービスＳを移動できないとき、中程度の負荷のサービスＳを選択し（ＳＴ８）、前述同様にサービスＳを移動可能か否かを判定する（ＳＴ９）。

【００５１】また、ステップＳＴ９において移動可能なときにはステップＳＴ７に行くが、中程度の負荷のサービスＳが移動不可のとき、低負荷のサービスＳを選択し（ＳＴ１０）、前述同様にサービスＳを移動可能か否か

を判定する（ＳＴ１１）。

【００５２】ステップＳＴ１１においても、移動可能なときにはステップＳＴ７に行くが、低負荷のサービスＳが移動不可のとき、ステップＳＴ１へ戻る。

（具体例１）次に、以上のような各ステップＳＴ１～ＳＴ１１において、１つのプロセスＰ１のみを有する１つのサービスＳ１の移動に際し、ＣＰＵ負荷のみを検討する場合について説明する。

【００５３】具体的には、図５に示す負荷状況において、ノードＮ１のＣＰＵ負荷値が９０％を越えた際に、サービスＳ１を移動させる場合の移動指示部４の動作を述べる。

【００５４】ステップＳＴ３において、ノードＮ１のＣＰＵ負荷は、サービスＳ１～Ｓ３を足して９０％であるため、ノードＮ１が過負荷と判定される。また、ステップＳＴ４において、ノードＮ１で最も高負荷のサービスＳ１が転送対象として選択される。

【００５５】次いで、ステップＳＴ５において、最低の負荷のノードＮ２が選択される。例えば、サービスＳ１を他ノードＮ２又はＮ３へ移動した場合を仮定し、ノードＮ２，Ｎ３にてサービスＳ１を実行する場合のＣＰＵ負荷値を試算する。その試算結果は、ノードＮ２が２５％（＝５０％＊（１／２００）／（１／１００））であり、ノードＮ３が１６％（＝５０％＊（１／３００）／（１／１００））である。

【００５６】ここで、サービスＳ１を移動すると、最終的なＣＰＵ負荷値は、ノードＮ２では３５％（＝１０％＋２５％）となり、ノードＮ３では４６％（＝３０％＋１６％）となる。従って、最終的なＣＰＵ負荷値の小さいノードＮ２は、ステップＳＴ５により最低の負荷のノードＮ２として選択され、ステップＳＴ６の（２）式により（先ＣＰＵidol９０％＞２５％）移動可能と判定され、ステップＳＴ７によりサービスＳ１が移動される。

（具体例２）また、具体例と同一のＣＰＵ性能値において、他のサービスが移動される場合について説明する。図６に示す負荷状況において、サービスＳ１の移動を仮定した場合、各ノードＮ２，Ｎ３の負荷は、ノードＮ２が１１０％（＝８５％＋２５％）となり、ノードＮ３が９１％（＝７５％＋１６％）となる。この場合、ノードＮ２，Ｎ３の負荷が高いので、サービスＳ１を移動できない。

【００５７】よって、中程度の負荷であるサービスＳ２の移動を検討する。サービスＳ２を移動した場合の計算は、ノードＮ２が１００％（＝８５％＋１５％）となり、ノードＮ３が８５％（＝７５％＋１０％）となるので、サービスＳ２をノードＣへ移動させる。

【００５８】このように負荷が高い順から、サービスＳ１，…の移動を計算し、移動可能なノードＮ３にサービスを移動させる。但し、全てのサービスＳ１～Ｓ３が移動不可能（他の全ノードＮ２，Ｎ３が高負荷状態）のとき、サービスＳ１～Ｓ３の移動をあきらめる。

【００５９】このように、ノードＮ１では、ノードＮ１自体又はプロセスＰｋにて障害発生あるいは高負荷の発生により、プロセスＰｋの実行が困難になると、（高負荷の発生時には予め当該プロセスＰｋを停止させた後、）低負荷の例えばノードＮ２にプロセス移動を指示して負荷分散を実行する。ノードＮ２では、プロセスの再開を実行する。

【００６０】ノードＮ２は、再開の実行時に、プロセスＰｋのmain（プログラムとしてのスタート）をフェッチし、チェックポイントＣＰのログからスタック積上げ／レジスタ情報設定等を実行し、ノードＮ１で中止されたプロセスＰｋを最新のチェックポイントＣＰ時点から再開して実行する。

【００６１】上述したように本実施形態によれば、各ノードＮ１～Ｎｎの現在の負荷値が記憶される負荷テーブル１と、各ノードＮ１～Ｎｎの処理性能値が記憶される性能値テーブル３とを有し、移動指示部４が、負荷テーブル１を参照して自ノードの過負荷を検出したとき、性能値テーブル３を参照して負荷を増加可能な移動先ノードを選択し、負荷移動指示を生成し、負荷分散部５が、移動指示部４により生成された負荷移動指示に基づいて、自ノードの負荷を所定単位毎に移動先ノードに移動させるので、従来とは異なり、メモリリソースに無駄を生じさせず、負荷集中時や分散要求時に負荷分散を実行させることができる。

【００６２】また、プロセスの実行中の移動により、サービスの継続性を保ちつつ、プログラミングによる負荷分散の意識をせずに、分散ノード間での負荷分散システムを構築することができる。

（第２の実施形態）次に、本発明の第２の実施形態に係る分散ノード間負荷分散方式の適用された計算機システムについて説明する。

【００６３】本実施形態は、第１の実施形態中、各ノードＮ１～Ｎｎの平均の負荷よりも高負荷のサービスＳがある場合、この高負荷のサービスＳのみが各ノードＮ１～Ｎｎを順番に移動する（たらい回しされる）場合があることを考慮し、このたらい回し動作の阻止を図るものである。

【００６４】具体的には、移動指示部４は、前述した機能に加え、図７に示すように、例えば各サービスＳ１～Ｓ９が負荷の程度に応じて配列される高、中、低の３段階のキューＱ１～Ｑ３を有し、各段階の負荷のサービスを選択（ＳＴ４，８，１０）する際に、各段階のキューの先頭にあるサービスＳ７（Ｓ８又はＳ９）を移動対象として選択する機能と、サービスＳ７（Ｓ８又はＳ９）が移動されたときにはこのサービスＳ７（Ｓ８又はＳ９）を該当する段階のキューＱ１（Ｑ２又はＱ３）の末尾に接続する機能とをもっている。

【００６５】なお、図７中のサービスＳの添字及び各Ｑ

１～Ｑ３内のサービスＳの個数は、単なる一例であり、適宜変更可能なことは言うまでもない。次に、以上のように構成された計算機システムの動作を説明する。なお、この説明は、第１の実施形態と比較して述べる。

【００６６】前述した第１の実施形態の場合、図５と同一のＣＰＵ性能値のノードＮ１において、図８に示すＣＰＵ負荷状況であるとする。この場合、ノードＮ１は、サービスＳ１をノードＮ２に移動させる。ここで、サービスＳ１のＣＰＵ負荷値が３５％～４５％の範囲内で上下すると、ノードＮ２は、サービスＳ１を他のノードＮ３に移動させる可能性がある。また、ノードＮ３はサービスＳ１をさらに他のノードＮ４に移動させる。以下同様に、サービス１のみが各ノードＮ５～Ｎｎを順番に移動する可能性がある。

【００６７】一方、本実施形態では、サービス選択用のキューを設けた構成により、移動対象のサービスＳ１～Ｓ９が図９に示すようにキューＱ１～Ｑ３に接続される。

【００６８】ここで、ノードＮ１がキューＱ１の先頭のサービスＳ１をノードＮ２に移動させると、ノードＮ２では、図１０に示すように、このサービスＳ１がキューＱ１の最後に接続される。

【００６９】これにより、ノードＮ２がサービスＳを移動させる場合、高負荷のキューＱ１の先頭であるサービスＳ６が移動対象となる。従って、第１の実施形態とは異なり、サービスＳ１のみが順番に移動するたらい回し動作を阻止することができる。

【００７０】上述したように本実施形態によれば、第１の実施形態の効果に加え、あるサービス（例えば全ノード中で一番負荷の高いサービス）のみがたらい回しにされる動作を阻止することができる。

（第３の実施形態）次に、本発明の第３の実施形態に係る分散ノード間負荷分散方式の適用された計算機システムについて説明する。

【００７１】本実施形態は、第１の実施形態中、通常は低負荷で一時的に高負荷になるが直ぐに低負荷に復帰するサービスＳがある場合、このサービスＳを移動させる場合があることを考慮し、この一時的に高負荷となるサービスＳの移動の阻止を図るものである。

【００７２】具体的には、負荷テーブル１は、前述した現在の負荷状況に加え、過去の負荷状況の平均値及び平均２乗誤差が記憶されるものである。

【００７３】負荷管理部２は、前述した機能に加え、過去の負荷状況の平均値及び平均２乗誤差を負荷テーブル１に書込む機能をもっている。

【００７４】移動指示部４は、前述した機能に加え、ステップＳＴ４でノードが過負荷か否かを判定する際に、現在の負荷値と負荷値の平均値とが著しく離れており、且つ平均２乗誤差が小さいノードＮを選択しない機能を有している。

【００７５】次に、以上のように構成された計算機システムの動作を説明する。なお、この説明は、第１の実施形態と比較して述べる。

【００７６】いま、図１１に示すように、通常は低負荷で一瞬だけ高負荷になるサービスＳ１があるとする。このサービスＳ１は、一瞬だけ負荷が上昇したが、通常は低負荷であるので、サービスＳ１を移動せずに時間の経過を待つ方がよい。

【００７７】しかし、第１実施形態では、高負荷となる時間Ａにおいて、移動対象のサービスＳ１を選択する場合、このサービスＳ１を移動対象とする可能性がある。

【００７８】一方、本実施形態では、負荷の平均値と平均２乗誤差とを管理する構成により、移動対象のサービスＳｋを選択する際に、現在の負荷と負荷の平均値とが著しく離れており、且つ平均２乗誤差が小さいサービスＳ１を移動対象に選択しない。

【００７９】これにより、通常は低負荷で一時的に高負荷となるサービスＳ１の移動を阻止できるので、移動によるオーバヘッドを抑制することができる。

【００８０】上述したように本実施形態によれば、第１の実施形態の効果に加え、通常は低負荷値で一時的に高負荷値となる負荷の移動を阻止できるので、移動によるオーバヘッドを抑制することができる。

（第４の実施形態）次に、本発明の第４の実施形態に係る分散ノード間負荷分散方式の適用された計算機システムについて説明する。

【００８１】本実施形態は、第１の実施形態中、多数のノードＮ１～Ｎｎを有する計算機システムの場合、最低の負荷をもつノードＮｉを選択する際に、ノード数ｎに比例してノードＮ１～Ｎｎの負荷を算出する処理のオーバヘッドを増大させることを考慮し、このオーバヘッドの抑制を図るものである。

【００８２】具体的には、移動指示部４は、前述した機能に加え、移動対象の例えばノードＮ１～Ｎ４を順番に配列したキューＱｍを有し、前述したステップＳＴ５によるノード選択の際に、最低の負荷のノードＮ２を選択するのではなく、キューＱｍの先頭にあるノードＮｓを移動先として選択する機能と、キューＱｍの先頭のノードＮにサービスＳを移動可能なとき、サービスＳをその先頭のノードＮに移動させる機能と、サービスＳを移動させたノードＮをキューＱｍの末尾に接続する機能とを有している。

【００８３】次に、以上のように構成された計算機システムの動作を説明する。

【００８４】本実施形態では、ノード選択用のキューＱを設けた構成により、移動先の候補としてノードＮ１～Ｎｘが待ち行列に接続される。

【００８５】例えば、ノードＮ１→ノードＮ２→ノードＮ３→ノードＮ４というキューＱｍがあり、ノードＮ２から高負荷のサービスＳ１をノードＮ１に移動するとす

る。

【００８６】ここで、移動指示部４は、ノード選択の際に、キューＱｍの先頭にあるノードＮ１を移動先として選択し、そのノードＮ１の負荷を算出してそのノードＮ１にサービスＳを移動可能なとき、サービスＳをそのノードＮ１に移動させる。

【００８７】また、移動指示部４は、サービスＳの移動により負荷の増えたノードＮ１をキューＱｍの末尾に接続する一方、サービスＳを移動して負荷の減ったノードＮ２をキューＱｍの先頭に接続する。サービス移動後のキューＱｍの状態は、ノードＮ１→ノードＮ３→ノードＮ４→ノードＮ２のようになる。

【００８８】このように、負荷を移動したノードがキューの先頭へ接続され、負荷の移動されたノードはキューの最後に接続されることにより、低負荷のノードがキューＱｍの先頭へ配置され、高負荷のノードがキューＱｍの後半に配置される。

【００８９】従って、次回、負荷移動先を算出する場合もキューＱｍの先頭からのヒット率が高くなり、全ノードＮ１～Ｎｎの負荷を算出する場合に比べ、オーバヘッドを抑制することができる。

【００９０】上述したように本実施形態によれば、第１の実施形態の効果に加え、ノードＮ２がサービスＳの移動先を選択する際に、他の全ノードＮ１，Ｎ３～Ｎｎのうち、一部のノードを検索すればよいので、サービス移動のオーバヘッドを抑制することができる。

【００９１】なお、上記各第２～第４の実施形態は、第１の実施形態に個別に適用した場合を説明したが、これに限らず、適宜組合せて同時に適用する構成としても、本発明を同様に実施して同様の効果を得ることができる。

【００９２】また、上記実施形態に記載した手法は、コンピュータに実行させることのできるプログラムとして、磁気ディスク（フロッピーディスク、ハードディスクなど）、光ディスク（ＣＤ－ＲＯＭ、ＤＶＤなど）、光磁気ディスク（ＭＯ）、半導体メモリなどの記憶媒体に格納して頒布することもできる。

【００９３】その他、本発明はその要旨を逸脱しない範囲で種々変形して実施できる。

【００９４】

【発明の効果】以上説明したように本発明によれば、メモリリソースに無駄を生じさせず、負荷集中時や分散要求時に負荷分散を実行できる分散ノード間負荷分散方式を提供できる。

【図面の簡単な説明】

【図１】本発明の第１の実施形態に係る分散ノード間負荷分散方式の適用された計算機システムの構成を示す模式図

【図２】同実施形態における負荷テーブルの構成を示す模式図

【図３】同実施形態における動作を説明するための模式図

【図４】同実施形態における移動指示部の動作を説明するためのフローチャート

【図５】同実施形態における動作を説明するための負荷状況を示す模式図

【図６】同実施形態における動作を説明するための負荷状況を示す模式図

【図７】本発明の第２の実施形態に係る分散ノード間負荷分散方式に用いられるキューの内容を示す模式図

【図８】同実施形態における動作を説明するための負荷状況を示す模式図

【図９】同実施形態における動作を説明するためのキューの内容を示す模式図

【図１０】同実施形態における動作を説明するためのキューの内容を示す模式図

【図１１】本発明の第３の実施形態に係る分散ノード間負荷分散方式を説明するためのサービスの負荷値を示す模式図

【符号の説明】

１…負荷テーブル
２…負荷管理部
３…性能値テーブル
４…移動指示部
５…負荷分散部
６…プロセスファイル
７…バックアップファイル
８…ジャケットルーチン
９…デーモン
９ａ…未確定キュー
９ｂ…確定キュー
Ｎ１～Ｎｎ…ノード
Ｐ１～Ｐｍ…プロセス

【図２】

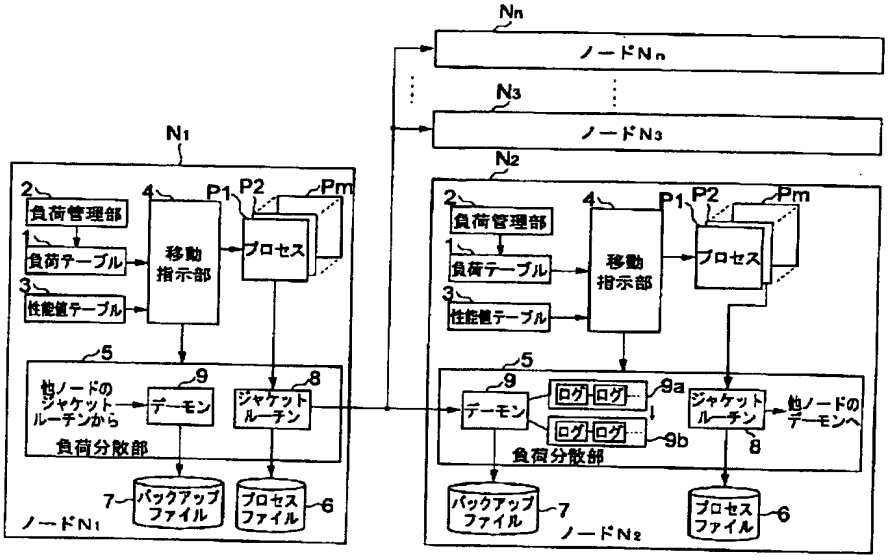
ノード	サービス	プロセス	ＣＰＵ負荷	メモリ負荷	ディスク負荷
ノードN1	サービスS1	プロセスP1			
		プロセスP2			
	サービスS2	プロセスP3			
		プロセスP4			
ノードN2	サービスS3	プロセスP5			

【図６】

1

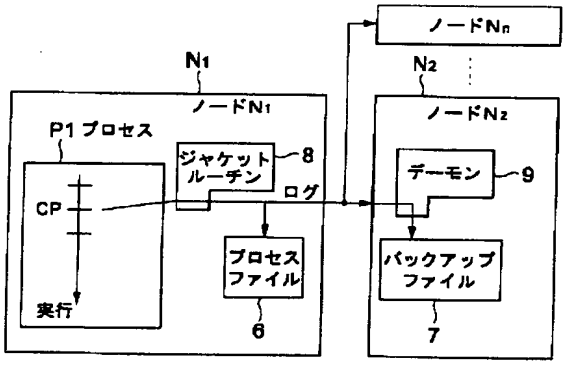
ノードN1	ノードN2	ノードN3
サービスS1 50% サービスS2 30% サービスS3 10%	サービスS4 85%	サービスS5 75%

【図１】

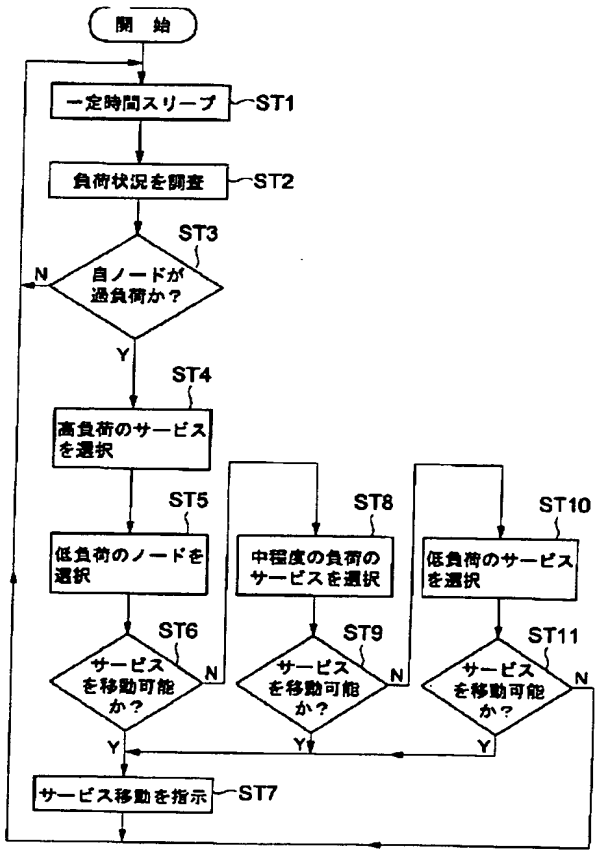


【図３】

【図４】



【図８】



ノードN1	ノードN2	ノードN3
サービスS1 35% サービスS2 30%	サービスS6 30%	
サービスS3 20% サービスS4 10%	サービスS7 20%	サービスS8 28% サービスS9 20%
サービスS5 5%		

【図９】

ノードN1	ノードN2	ノードN3
高負荷のキューQ1 サービスS2→サービスS1	サービスS6	
中負荷のキューQ2 サービスS3→サービスS4	サービスS7	サービスS8→サービスS9
低負荷のキューQ3 サービスS5		

【図５】

		ノードN1	ノードN2	ノードN3
1	負荷テーブル	サービス CPU負荷	サービス CPU負荷	サービス CPU負荷
		サービスS1 50% サービスS2 30% サービスS3 10%	サービスS4 10%	サービスS5 30%
3	性能値テーブル	CPU性能（100MHz 1個）	CPU性能（100MHz 2個）	CPU性能（300MHz 1個）
		1/100	1/200	1/300

【図７】

高負荷のキュー Q1	サービスS1→サービスS4→サービスS7
中程度の負荷のキュー Q2	サービスS2→サービスS5→サービスS8
低負荷のキュー Q3	サービスS3→サービスS6→サービスS9

【図１０】

ノードN1	ノードN2	ノードN3
高負荷のキューQ1 サービスS2	サービスS1→サービスS6	
中負荷のキューQ2 サービスS3→サービスS4	サービスS7	サービスS8→サービスS9
低負荷のキューQ3 サービスS5		

【図１１】